



Mapa e conexão com a API

App de posto de gasolina Mapa

Agora vamos aproveitar o código criado na aula 13 para criar o mapa com os postos de gasolina, primeiramente vamos somente mostrar o mapa e posteriormente vamos pegar a lista de postos da API Json server.

Vamos alterar nosso componente tab1 para adicionar o código da aula 13.

```
export class Tab1Page {  
  
  @ViewChild('map') mapRef!: ElementRef<HTMLDivElement>;  
  
  newMap!: GoogleMap;  
  
  center: any = {  
  
    lat: -23.0008826,  
  
    lng: -43.3505312,  
  
  };  
  
  markerId: string = '';  
  
  constructor() {
```



```
console.log(this.mapRef)

}

ionViewDidEnter() {

  this.createMap();

}

async locate() {

  if(this.newMap) await this.newMap.enableCurrentLocation(true);

}

async createMap() {

  try {

    this.newMap = await GoogleMap.create({

      id: 'capacitor-google-maps',

      element: this.mapRef.nativeElement,

      apiKey: environment.google_maps_api_key,

      config: {

        center: this.center,

        zoom: 13,

      },
```



```
});  
  
console.log('newmap', this.newMap);  
  
await this.addMarker(this.center.lat, this.center.lng);  
  
await this.addListeners();  
  
} catch(e) {  
  
    console.log(e);  
  
}  
  
}  
  
async setCamera() {  
  
    // Move the map programmatically  
  
    await this.newMap.setCamera({  
  
        coordinate: {  
  
            lat: -23.0008826,  
  
            lng: -43.3505312,  
  
        },  
  
        zoom: 13,  
  
        // animate: true  
  
    });
```



```
// Enable traffic Layer
```

```
await this.newMap.enableTrafficLayer(true);
```

```
if(Capacitor.getPlatform() !== 'web') {
```

```
    await this.newMap.enableIndoorMaps(true);
```

```
    // await this.newMap.setMapType(MapType.Satellite);
```

```
}
```

```
await this.newMap.setPadding({
```

```
    top: 50,
```

```
    left: 50,
```

```
    right: 0,
```

```
    bottom: 0,
```

```
});
```

```
}
```

```
async addMarkers(lat:number, lng:number, title:string) {
```

```
    await this.newMap.addMarkers([
```

```
        {
```

```
            coordinate: {
```

```
                lat: lat,
```



```
    lng: lng,  
  },  
  
  title: title ,  
  
  draggable: true  
}  
  
]);  
  
}  
  
async addMarker(lat:number, lng: number) {  
  
  // Add a marker to the map  
  
  if(this.markerId) this.removeMarker();  
  
  this.markerId = await this.newMap.addMarker({  
  
    coordinate: {  
  
      lat: lat,  
  
      lng: lng,  
  
    },  
  
    title: 'Posto de Gasolina',  
  
    draggable: true,
```



```
});  
  
}  
  
async removeMarker(id?:any) {  
  
    await this.newMap.removeMarker(id ? id : this.markerId);  
  
}  
  
async addListeners() {  
  
    // Handle marker click  
  
    await this.newMap.setOnMarkerClickListener((event) => {  
  
        console.log('setOnMarkerClickListener', event);  
  
        this.removeMarker(event.markerId);  
  
    });  
  
    await this.newMap.setOnCameraMoveStartedListener((event) => {  
  
        console.log(event);  
  
    });  
  
    await this.newMap.setOnCameraIdleListener((event) => {  
  
        console.log('idle: ', event);  
  
        // alert(event);  
  
        this.center = {
```



```
lat: event.latitude,  
  
lng: event.longitude  
  
}  
  
this.addMarker(this.center.lat, this.center.lng);  
  
});  
  
await this.newMap.setOnMapClickListener((event) => {  
  
    console.log('setOnMapClickListener', event);  
  
    this.addMarker(event.latitude, event.longitude);  
  
});  
  
await  
this.newMap.setOnMyLocationButtonClickListener((event:any) => {  
  
    console.log('setOnMyLocationButtonClickListener', event);  
  
    this.addMarker(event.latitude, event.longitude);  
  
});  
  
await this.newMap.setOnMyLocationClickListener((event) => {  
  
    console.log('setOnMyLocationClickListener', event);  
  
    this.addMarker(event.latitude, event.longitude);  
  
});
```

}



}

Vamos fazer algumas alterações no código do mapa, porém antes disso vamos adicionar ao Json Server uma lista de postos de gasolina para adicionarmos no mapa.

```
gasStations: GasStation[] = [];
```

```
constructor(private service: GasService) {}
```

```
ngOnInit(): void {
```

```
  this.getGasStations();
```

```
}
```

```
getGasStations(): void {
```

```
  this.service.getGasStation().subscribe(
```

```
  {
```

```
    next: (response) => {
```

```
      console.log('entrou no response')
```

```
      console.log(response)
```

```
      this.gasStations = response;
```

```
    },
```

```
    error: (erro: any) => {
```



```
        console.log('entrou no erro')  
  
        console.log(erro)  
  
    }  
  
}  
  
)  
  
}
```



Agora vamos adicionar os Markers no mapa com os postos de gasolina.

```
async loadMarkers() {  
  
    if(this.newMap) await  
  
    this.gasStations  
  
    .forEach(gasStation => {  
  
        this.addMarkers(gasStation.lat,gasStation.lng, gasStation.title);  
  
    });  
  
}
```

Deploy da aplicação e lojas

Cada tipo de sistema operacional tem sua própria loja, o Android tem Play Store que você pode publicar os aplicativos de forma gratuita e

iOS tem App Store que você tem que pagar 100 dólares anuais para conseguir fazer a publicação. 

Tendo como base essas informações vamos seguir um exemplo de publicação usando o Android na Play Store.

Basicamente você deve entrar no Android Studio gerar uma versão assinada do aplicativo e posteriormente após cadastrar com distribuidor no google adicionar para revisão com uma versão Release.

Existem vários tipos de versão na loja, podemos criar por exemplo uma versão para teste interno ou externo.

Na App Store da Apple funciona mais ou menos da mesma maneira entrar no Xcode e gerar uma publicação assinada do aplicativo. Durante as vídeo aulas vamos mostrar como isso funciona.

Seguindo as instruções da documentação oficial do Ionic vamos verificar com mais detalhes esse processo através da URL a seguir:

<https://ionicframework.com/docs/deployment/play-store>

Atividade Extra

Para se aprofundar no assunto desta aula, assista o vídeo:

Canal: Valdir Cezar Tutoriais

Título a ser buscado no youtube: **Aula 15 - Exceção personalizada - MEU PRIMEIRO PROJETO FULL STACK**



Referência Bibliográfica

ANGULAR DOCS. **Angular IO**. Disponível em: <<https://angular.io>>. Acesso em: 30 nov. 2022.

IONIC FRAMEWORK. **IONIC**. Disponível em: <<https://ionicframework.com>>. Acesso em: 30 nov. 2022.

CAPACITOR FRAMEWORK. **CAPACITOR**. Disponível em: <<https://capacitorjs.com/docs>>. Acesso em: 30 nov. 2022.

Atividade Prática 16 - Criando seu próprio App de Mapas

Título da Prática: Criação de exceções.

Objetivos: Praticar a programação de criação de projetos em Ionic

Materiais, Métodos e Ferramentas: Iremos fazer uma pesquisa na internet e utilizar os conceitos de tipos de exceções para criar a nossa prática.

Atividade Prática

Criar um aplicativo Ionic capacitor que utiliza mapas. Pode ser lista de igrejas ou de faculdades em sua cidade.



Gabarito Atividade Prática

Você pode utilizar o Json server para criar uma lista de Igrejas ou Faculdades e mostrar os detalhes no mapa através do modal use a criatividade.

Primeiro criamos o HTML e adicionamos o mapa

```
<ion-content [fullscreen]="true">
```

```
<div class="maps">
```

```
<capacitor-google-maps #map></capacitor-google-maps>
```

```
</div>
```

```
</ion-content>
```

Agora criamos o arquivo de service

```
import { Injectable } from '@angular/core'
```

```
import { HttpClient, HttpHeaders } from '@angular/common/http'
```

```
import { Observable } from 'rxjs'
```

```
import { Church } from '../models/church'
```

```
const httpOptions = {
```

```
headers: new HttpHeaders { 'Content-Type': 'application/json' }
```



```
@Injectable({
```

```
providedIn: 'root'
```

```
export class ChurchService {
```

```
BASE_URL: string = 'http://localhost:3000/'
```

```
constructor(private http: HttpClient) {}
```

```
/** GET gas api */
```

```
getChurches(): Observable<Church[]> {
```

```
var url: string = this.BASE_URL + 'churches'
```

```
console.log(url)
```

```
return this.http.get<Church[]>(url)
```

E posteriormente vamos modificar nosso componente

```
import { Component, ElementRef, ViewChild } from '@angular/core'
```



```
import { Capacitor } from '@capacitor/core'
```

```
import { GoogleMap, MapType } from '@capacitor/google-maps'
```

```
import { ModalController } from '@ionic/angular'
```

```
import { environment } from '../environments/environment'
```

```
import { ModalPage } from '../modal/modal.page'
```

```
import { Church } from '../models/church'
```

```
import { ChurchService } from '../services/church.service'
```

```
import { GasService } from '../services/gas.service'
```

```
@Component({
```

```
  selector: 'app-tab1',
```

```
  templateUrl: 'tab1.page.html',
```

```
  styleUrls: ['tab1.page.scss']
```

```
export class Tab1Page {
```

```
  @ViewChild('map') mapRef: ElementRef<HTMLDivElement>
```

```
  newMap!: GoogleMap
```

```
  center: any = {
```

lat: -23.0008826



lng: -43.3505312

markerId: string = "

churchs: Church[] = []

constructor()

public modalCtrl: ModalController

private service: ChurchService

ngOnInit(): void

this.getGasStations()

getGasStations(): void

this.service.getChurchs().subscribe()

next: (response) =>

console.log(response)

this.churchs = response

```
error: (erro: any) => |
```



```
console.log('entrou no erro'
```

```
console.log(erro)
```

```
ionViewDidEnter() {
```

```
  this.createMap()
```

```
async locate() {
```

```
  if (this.newMap) await this.newMap.isReady().then(() => {
```

```
async createMap() {
```

```
  try {
```

```
    this.newMap = await GoogleMap.create({
```

```
      id: 'capacitor-google-maps',
```

```
      element: this.mapRef.nativeElement
```



```
apiKey: environment.google_maps_api_key
```



```
config:
```

```
center: this.center
```

```
zoom: 13
```

```
await this.addMarker(this.center.lat, this.center.lng
```

```
await this
```

```
//await this.locate();
```

```
await this
```

```
catch (e) {
```

```
console.log(e)
```

```
async
```

```
if (this.newMap) await
```

```
this.churchs
```

```
church =>
```

```
this.addMarkers(church.lat, church.lng, church.name)
```



```
async setCamera()
```

```
// Move the map programmatically e coloca na posição desejada
```

```
await this.newMap.setCamera
```

```
coordinate:
```

```
lat: -23.0008826,
```

```
lng: -43.3505312
```

```
zoom: 13
```

```
// Enable traffic Layer
```

```
await this.newMap.enableTrafficLayer(true)
```

```
if (Capacitor.getPlatform() !== 'web') {
```

```
await this.newMap.enableIndoorMaps(true)
```

```
await this.newMap.setPadding
```



top: 50

left: 50

right: 0

bottom: 0

```
async addMarker(lat: number, lng: number, title: string) {
```

```
  await this.newMap.addMarker({
```

```
    coordinate: {
```

```
      lat: lat,
```

```
      lng: lng,
```

```
    },
```

```
  })
```

```
  title: title
```

```
})
```

```
async addMarker(lat: number, lng: number, title: string) {
```



```
// Add a marker to the map
```

```
if (this.markerId) this.removeMarker();
```

```
this.markerId = await this.newMap.addMarker({
```

```
  coordinate: {
```

```
    lat: lat,
```

```
    lng: lng,
```

```
  },
```

```
  title: 'Posto de Gasolina',
```

```
  draggable: true,
```

```
});
```

```
}
```

```
async removeMarker(id?: any) {
```

```
  await this.newMap.removeMarker(id ? id : this.markerId);
```

```
}
```

```
async addListeners() {
```

```
  // Handle marker click
```

```
  await this.newMap.setOnMarkerClickListener((event) => {
```

```
    let id: number = Number(event.markerId);
```



```
console.log(id);

console.log(this.churchs[id]);

this.goToDetail(this.churchs[id])

});

await this.newMap.setOnCameraMoveStartedListener((event) => {

    console.log(event);

});

await this.newMap.setOnMapClickListener((event) => {

    console.log('setOnMapClickListener', event);

    this.addMarker(event.latitude, event.longitude);

});

await this.newMap.setOnMyLocationButtonClickListener((event:
any) => {

    console.log('setOnMyLocationButtonClickListener', event);

    this.addMarker(event.latitude, event.longitude);

});

await this.newMap.setOnMyLocationClickListener((event) => {

    console.log('setOnMyLocationClickListener', event);

    this.addMarker(event.latitude, event.longitude);
```



```
});  
  
}  
  
async goToDetail(church: Church) {  
  
    //this.churchs[this.indexChurch]  
  
    //sessionStorage.setItem("post",  
    JSON.stringify(this.churchs[this.indexChurch]));  
  
    const modal = await this.modalCtrl.create({  
  
        component: ModalPage,  
  
        cssClass: 'my-custom-modal-css',  
  
        componentProps: {  
  
            name: church.name,  
  
            lat: church.lat,  
  
            lng: church.lng,  
  
        },  
  
    });  
  
    modal.present();  
  
}  
  
}
```

Ir para exercício

